Interview Veryfi

Marcos Reyes 23/08/2022

Requirements

Veryfi's requirements

"You have the OCR output for a couple of receipts, and we need to extract some relevant information such as vendor name, vendor address, line items and total. It is intended for you to develop a generalized way to extract this information for both receipts, bearing in mind that any receipt with the same format should be supported"

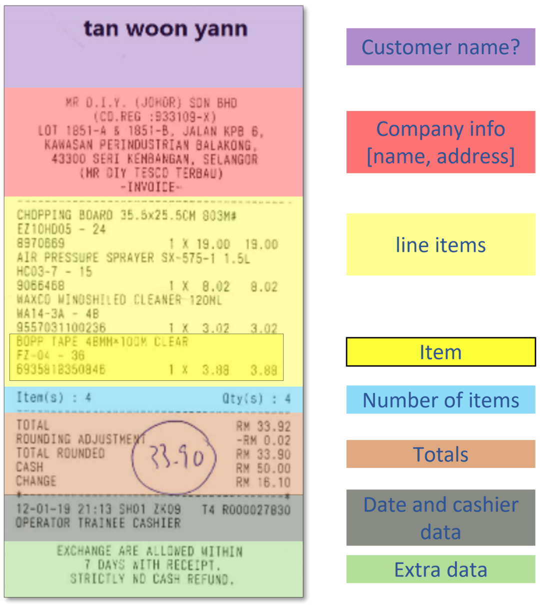
Summary:

1. Develop a generalized way to extract information from OCR output for a couple of receipts.
2. Extract some relevant information:
    a. vendor name
    b. vendor address
    c. line items
    d. total

research

input

The input files are text files (.txt) from images of receipts. This is the structure of receipts:



```
tan woon yann

MR D.I.Y. (JOHOR) SDN BHD
(CO.REG :933109-X)
LOT 1851-A & 1851-B, JALAN KPB 6,
KAWASAN PERINDUSTRIAN BALAKONG,
43300 SERI KEMBANGAN, SELANGOR
(MR DIY TESCO TERBAU)
-INVOICE-
-----------------------------------------
CHOPPING BOARD 35.5x25.5CM 803M#
EZ10HD05 - 24
8970669                1 X 19.00   19.00
AIR PRESSURE SPRAYER SX-575-1 1.5L
HC03-7 - 15
9066468                1 X  8.02    8.02
WAXCO WINDSHILED CLEANER 120ML
WA14-3A - 48
9557031100236          1 X  3.02    3.02
BOPP TAPE 48MM*100M CLEAR
FZ-04 - 36
6935818350846          1 X  3.88    3.88
Item(s) : 4                    Qty(s) : 4
-----------------------------------------
TOTAL                           RM 33.92
ROUNDING ADJUSTMENT            -RM 0.02
TOTAL ROUNDED        33.90      RM 33.90
CASH                            RM 50.00
CHANGE                          RM 16.10
*---------------------------------------*
12-01-19 21:13 SH01 ZK09   T4 R000027830
OPERATOR TRAINEE CASHIER
EXCHANGE ARE ALLOWED WITHIN
7 DAYS WITH RECEIPT.
STRICTLY NO CASH REFUND.
```

the (.txt) file is structured as:

<x1>,<y1>,<x2>,<y2>,<x3>,<y3>,<x4>,<y4>,<text>

Where “Xn” and “Yn” are positions (a box) on the receipt where there is a text. For example, the first line:

119,47,367,47,367,80,119,80,TAN WOON YANN

Output

Output must be a Javascript object notation (JSON) or a python dictionary with some relevant information

```
{
  "company": "MR D.I.Y. (JOHOR) SDN BHD",
  "date": "12-01-19",
  "address": "LOT 1851-A & 1851-B, JALAN KPB 6, KAWASAN PERINDUSTRIAN BALAKONG, 43300 SERI KEMBANGAN,
SELANGOR (MR DIY TESCO TERBAU)",

  "line_items":[
    {
      "sku":"8970669",
      "quantity":"1",
      "price":"19.00",
      "total":"19.00"
    },
    {
      "sku":"9066468",
      "quantity":"1",
      "price":"8.02",
      "total":"8.02"
    },
    {
      "sku":"9557031100236",
      "quantity":"1",
      "price":"3.02",
      "total":"3.02"
    },
    {
      "sku":"6935818350846",
      "quantity":"1",
      "price":"3.88",
      "total":"3.88"
    }
  ],
  "total": "33.90"
}
```

Relevant information comes from 4 different rereceipt’s structure:

1. Company info [name, address]
2. Date and cashier data
3. line items
4. Total

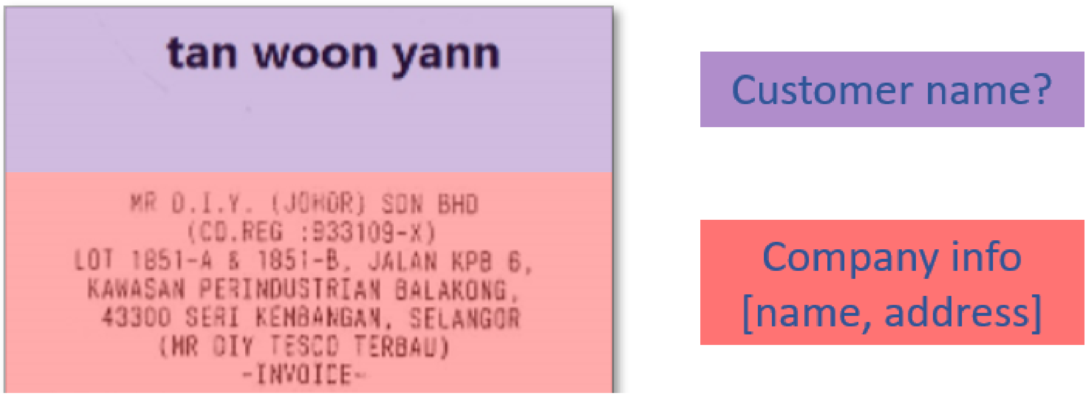
patterns

Here are output results, receipt image and (.txt) file input without “Xn” and “Yn” info.

Company info [name, address]

Company info starts at the beginning and finishes at line “-INVOICE-”

"company": "MR D.I.Y. (JOHOR) SDN BHD",
"address": "LOT 1851-A & 1851-B, JALAN KPB 6, KAWASAN PERINDUSTRIAN BALAKONG, 43300 SERI KEMBANGAN, SELANGOR (MR DIY TESCO TERBAU)",


tan woon yann

MR D.I.Y. (JOHOR) SDN BHD
(CO.REG :933109-X)
LOT 1851-A & 1851-B, JALAN KPB 6,
KAWASAN PERINDUSTRIAN BALAKONG,
43300 SERI KEMBANGAN, SELANGOR
(MR DIY TESCO TERBAU)
-INVOICE-

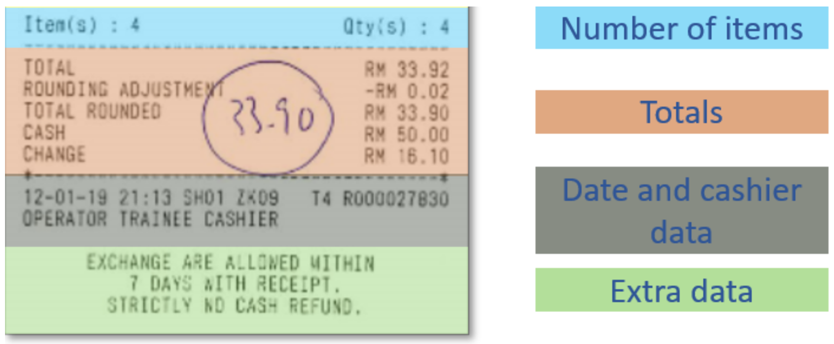
Customer name?
Company info [name, address]

TAN WOON YANN {no relevant, customer name, **first line**}
MR D.T.Y. (JOHOR) SDN BHD {relevant, company’s name, second line}
(CO.REG : 933109-X) {no relevant, company registration number, **third line, start with “CO.REG”**}
LOT 1851-A & 1851-B, JALAN KPB 6, {relevant, company’s address, between company registration number and “-invoice-” }
KAWASAN PERINDUSTRIAN BALAKONG, {relevant, company’s address}
43300 SERI KEMBANGAN, SELANGOR {relevant, company’s address}
(MR DIY TESCO TERBAU) {relevant, company’s address}
-INVOICE- {no relevant, label, **last line**}

Date and cashier data, and Total

This block starts at the “ITEM(S)” line and goes until the end of the document.

"date": "12-01-19",
"total": "33.90"



```
Item(s) : 4                    Qty(s) : 4
TOTAL                          RM 33.92
ROUNDING ADJUSTMENT            -RM 0.02
TOTAL ROUNDED        33.90     RM 33.90
CASH                           RM 50.00
CHANGE                         RM 16.10
12-01-19 21:13 SH01 ZK09   T4 R000027830
OPERATOR TRAINEE CASHIER
      EXCHANGE ARE ALLOWED WITHIN
         7 DAYS WITH RECEIPT.
       STRICTLY NO CASH REFUND.
```

Number of items
Totals
Date and cashier data
Extra data

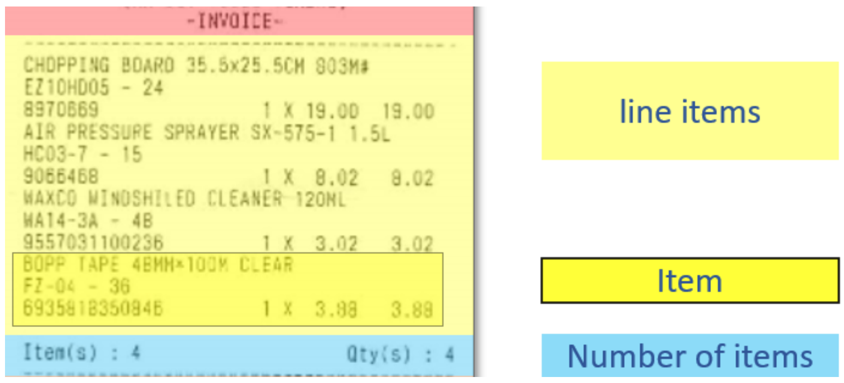
I could convert those lines into one string and use regex to find the total and date. Join lines with a special character like “,”.

ITEM(S) : 4
QTY(S) : 4
TOTAL
RM 33.92
ROUNDING ADJUSTMENT
-RM 0.02
TOTAL ROUNDED {relevant, it says where total is}
RM 33.90 {relevant, total value}
CASH
RM 50.00
CHANGE
RM 16.10
12-01-19 21:13 SH01 ZK09 {relevant, format date “dd-mm-yy”}
T4 R000027830
OPERATOR TRAINEE CASHIER
EXCHANGE ARE ALLOWED WITHIN
DAYS WITH RECEIPT.
STRICTLY NO CASH REFUND.

line items

This block starts at the “-INVOICE-” line and finishes at the “ITEM(S)” line.

```
"line_items":[
    {
       "sku":"8970669",
       "quantity":"1",
       "price":"19.00",
       "total":"19.00"
    },
    {
       "sku":"9066468",
       "quantity":"1",
       "price":"8.02",
       "total":"8.02"
    },
    {
       "sku":"9557031100236",
       "quantity":"1",
    "price":"3.02",
    "total":"3.02"
  },
  {
    "sku":"6935818350846",
    "quantity":"1",
    "price":"3.88",
    "total":"3.88"
  }
 ],
```



```
-INVOICE-
---------------------------------------------
CHOPPING BOARD 35.5x25.5CM 803M#
EZ10HD05 - 24
8970669                   1 X 19.00   19.00
AIR PRESSURE SPRAYER SX-575-1 1.5L
HC03-7 - 15
9066468                   1 X  8.02    8.02
WAXCO WINDSHILED CLEANER 120ML
WA14-3A - 48
9557031100236             1 X  3.02    3.02
BOPP TAPE 48MMx100M CLEAR
FZ-04 - 36
6935818350846             1 X  3.88    3.88
Item(s) : 4                      Qty(s) : 4
```

-INVOICE-
CHOPPING BOARD 35.5X25.5CM 803M#
EZ10HD05 - 24
8970669 { relevant, sku info, 2 lines before “X”}
1 { relevant, quantity info, 1 lines before “X”}
X { relevant, character reference, equal “X”}
19.00 { relevant, price info, 1 lines after“X”}
19.00 { relevant, total info, 2 lines after“X”}
AIR PRESSURE SPRAYER SX-575-1 1.5L
HC03-7 - 15
9066468 { relevant, sku info, 2 lines before “X”}
1 { relevant, quantity info, 1 lines before “X”}
X { relevant, character reference, equal “X”}
8.02 { relevant, price info, 1 lines after“X”}
8.02 { relevant, total info, 2 lines after“X”}
…
…
…
ITEM(S) : 4

Design

Functions

There are different functions needed to get and process data from (.txt) files to requirement output.

read_file(file_path)
1. check if the file exists
2. open file
3. check if it has the correct format (9 elements, int format for first 8 elements)
4. return file
5. close file

split_blok_by_index(start, end, list)
1. return list between index start and end

process_first_block(first_block) # first block without “-INVOICE-” line
1. find index of “CO.REG” line
2. Save the company’s name, line before “CO.REG” line
3. save the company's address, all line after “CO.REG” line.
    a. Join all lines with a “ “
    b. save it

process_third_block(third_block)
1. Join all lines together with a “,”
2. Find and save total (red) value with regex “,TOTAL ROUNDED,RM **33.90**,”
3. Find and save date using regex with this format “,dd-mm-yy HH:MM ”

process_second_block(second_block)
1. find all “X” index
2. save 2 lines before and after for each “X” index (sub objects)

find_index_of_character(block, element)
1. indices = [i for i, x in enumerate(block) if x == element]
2. return indices

Classes

Class Receipt:

1. Properties:
    a. list only text: **list**
    b. list all data: **list**
    c. main blocks: **list**
    d. company: **str**
    e. date: **str**
    f. address: **str**
    g. line items: **list [objects item]**
    h. total: **str**
2. Methods:
    a. __init__(file path)
        i. list only text, list all data: fill out
        ii. main blocks: fill out
        iii. fill_out_line_items
        iv. company
        v. date
        vi. address
    b. __fill_out_line_items(list_line_items)
    c. __build_result()
    d. result()
    e. __str__() (maybe)
    f. __index__()

Class Item:

1. Properties:
    a. sku: **str**
    b. quantity: **str**
    c. price: **str**
    d. total: **str**
2. methods:
    a. __init__(list_properties)
        i. sku = list_properties[0]
        ii. quantity
        iii. price
        iv. total
    b. __build_result()
    c. result()
    d. __str__()

Directory design

process receipt

1. **__init__.py**
2. **read_file.py**
3. **process.py**
4. **receipt.py**
5. GUI.py

main_basic.py
GUI_prototype.py